Machine Learning for Data Streams

Alexander Albrecht
WS 2019/20

Use Cases for Machine Learning & Data Streams

Sensor Processing

Process Monitoring

Location Tracking

Log Analysis

User Interaction

Market and Climate Prediction, etc.

Seminar - Learning Goals

Understand, implement, and deploy a challenging machine learning (ML) algorithm. (no optimization required)

Learn about state-of-the-art streaming techniques.

Build an ML-algorithm for data streams using Kafka and Kafka Streams.

Solve problems that arise from distributed computing.

Evaluate the quality and performance of your algorithm.

Write a scientific documentation.

Reveal new research questions for distributed computing (at best)

Seminar - Organization

Choose a research project (A, B, C or D).

Study the literature of your topic (books, papers, and online material).

Design a distributed algorithm with Kafka Streams that solves the problem of your projects.

Evaluate your solution w.r.t. accuracy/quality and performance.

Document your approach by writing a scientific documentation about as a GitHub page.

Seminar - Organization

Extent	4 SWS
Location	Campus II, Building F, Room F-E-06
Dates	Tuesday, 9:15 - 10:45 AM
Class	At most 8 participants (4 teams á 2 students)
Register	Informal email to alexander.albrecht@bakdata.com by October 18 (notification October 21)

Seminar - Registration Email

Add your distributed programming experience (lectures, seminars, some other courses, or projects).

Add a ranking of up to three research projects (A, B, C or D) that interest you (from the list shown today or own suggestions).

We do the final assignment in our first Kick-off meeting; so this is not a commit!

<optional> Add a team partner; you get either accepted or rejected together if seats get tight.

Seminar - Grading

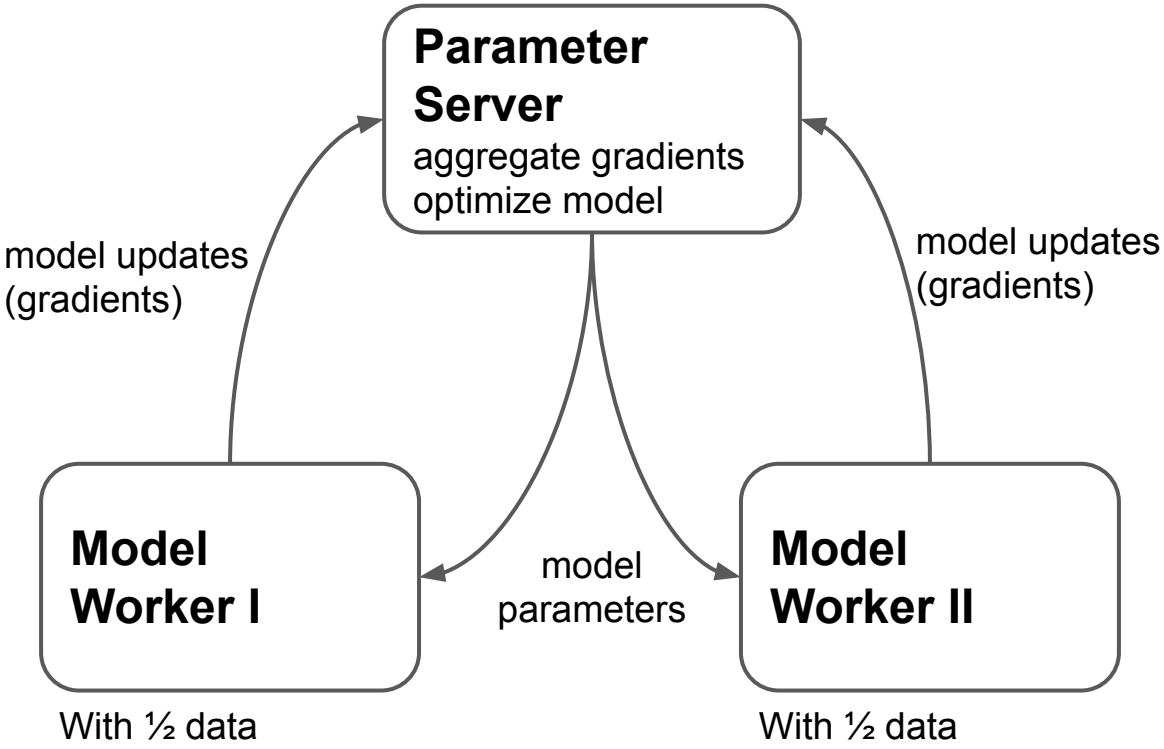
10%	Active participation during all seminar events.
00%	Regular meetings with advisor.
10%	Short presentation of the selected research paper.
15%	Intermediate presentation demonstrating insights regarding your research prototype.
15%	Final presentation demonstrating your solution.
20%	Implementation of a research prototype with Kafka and Kafka Streams (on GitHub).
30%	Documentation (on GitHub).

Project A
Distributed Model Training with Parameter Server

Machine Learning (ML) Algorithms

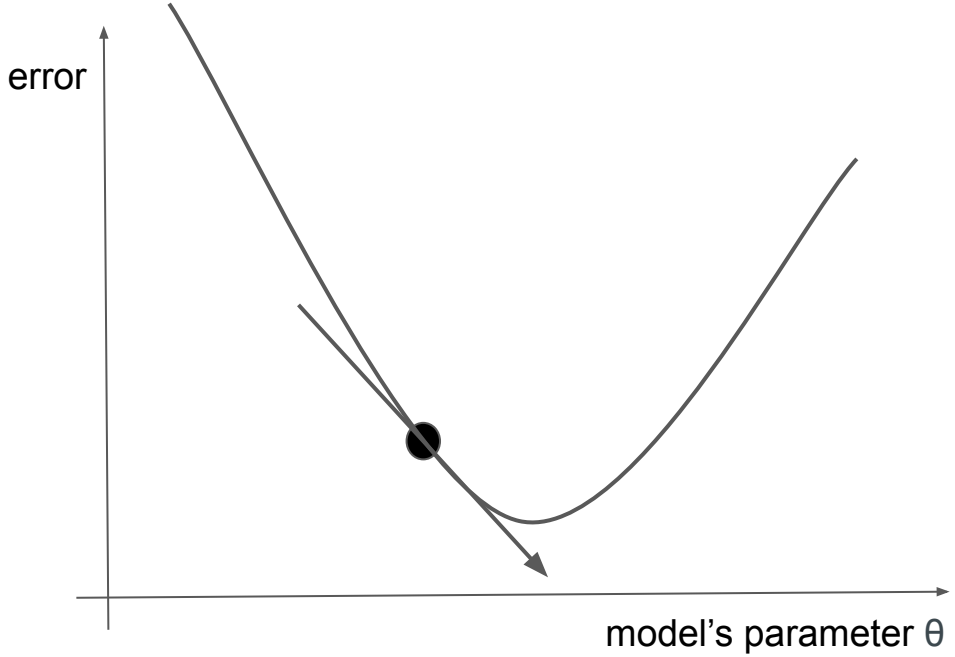
- Linear Regression
- Logistic Regression
- Support Vector Machine
- Clustering
- Neural Networks
- …

Learning ML Models



Modify Model's Parameters

Optimization method - (Stochastic) gradient descent: Follow the gradient of error w.r.t. to model's parameters improvement



Anatomy of Existing Parameter Server Systems

BSP Systems: Bulk Synchronous Parallel. One worker cannot continue to the next iteration until the Parameter Server receives all model updates and broadcasts a newly updated global parameter.

ASP Systems: Asynchronous Parallel. Workers proceed without waiting for each other, making ASP systems often faster than BSP systems in homogeneous clusters.

SSP Systems: Stale Synchronous Parallel. The fastest worker cannot exceed the slowest one more than a predefined number K of iterations.

Research Papers

Li, Mu, et al. "Scaling distributed machine learning with the parameter server." 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14). 2014.

Abadi, Martín, et al. "Tensorflow: A system for large-scale machine learning." 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16). 2016.

Jiang, Jiawei, et al. "Heterogeneity-aware distributed parameter servers." Proceedings of the 2017 ACM International Conference on Management of Data (SIGMOD '17). 2017.

Additional Readings

Hogwild!? Implementing Async SGD in Python

Scott Rome, November 18, 2017

How to Implement Linear Regression with Stochastic Gradient Descent from Scratch with Python

Jason Brownlee, October 28, 2016

Implementing Minibatch Gradient Descent for Neural Networks

Agustinus Kristiadi, June 21, 2016

An overview of gradient descent optimization algorithms

Sebastian Ruder, January 19, 2016

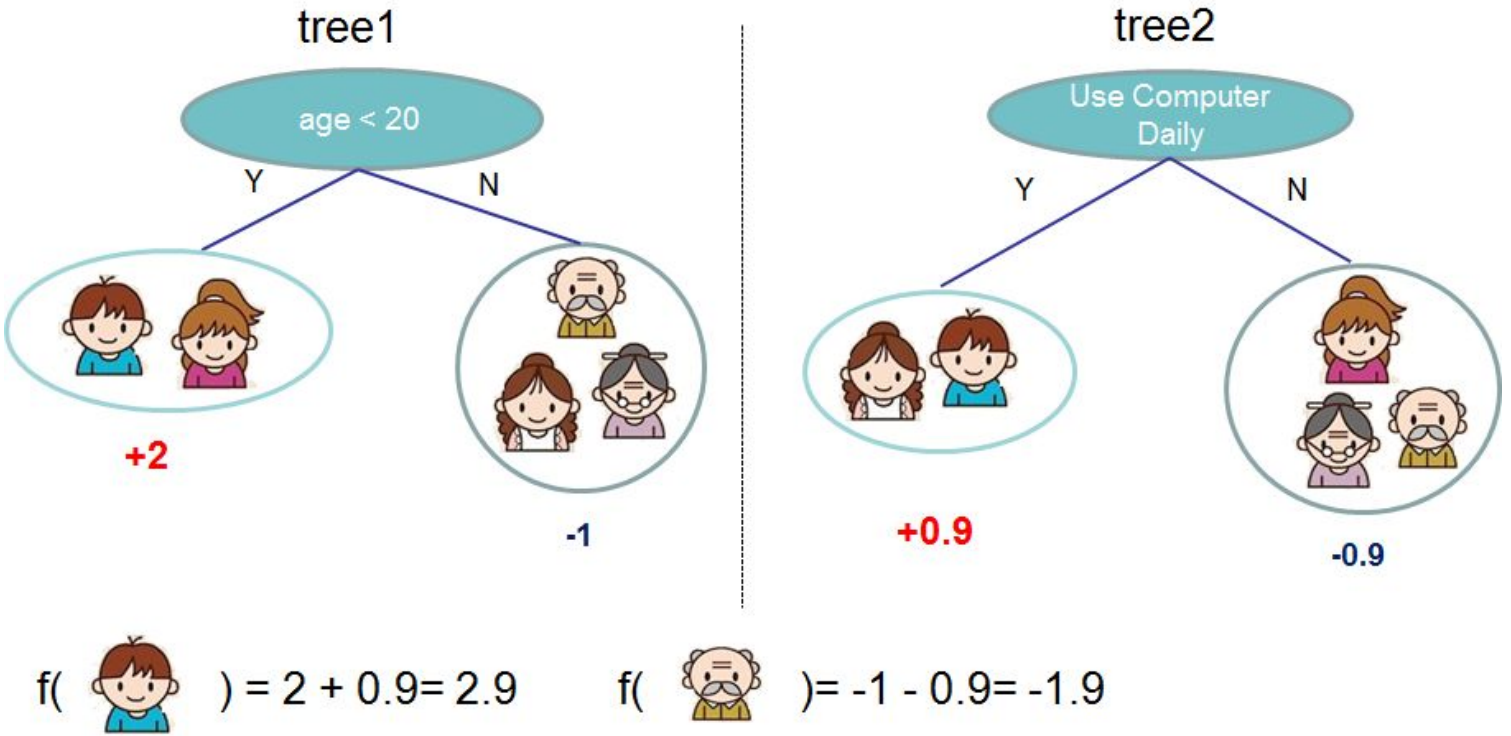
Project B
Gradient Boosting Decision Tree

Gradient Boosting Decision Tree

Gradient boosting tree (GBDT) is one of the most preferred choices in data analytics competitions such as Kaggle and KDDCup

Introduction to Boosted Trees, https://xgboost.readthedocs.io/en/latest/tutorials/model.html

Illustration of GBDT



Distributed implementations of GBDT

1. The training instances are partitioned onto a set of workers.
2. To split one tree node, each worker computes the gradient statistics of the instances. For each feature, an individual gradient histogram needs to be built.
3. A coordinator aggregates the gradient histograms of all workers, and finds the best split feature and split value.
4. The coordinator broadcasts the split result. Each worker splits the current tree node, and proceeds to new tree nodes

Research Papers

Fu, Fangeheng, et al. "An experimental evaluation of large scale gbdt systems." Proceedings of the VLDB Endowment 12.11 (2019): 1357-1370.

Jiang, Jie, et al. "TencentBoost: a gradient boosting tree system with parameter server." 2017 IEEE 33rd International Conference on Data Engineering (ICDE). IEEE, 2017.

Ponomareva, Natalia, et al. "Tf boosted trees: A scalable tensorflow based framework for gradient boosting." Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, Cham, 2017.

Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, 2016.

Project C
Collaborative Filtering: Alternating Least Squares (ALS)

Collaborative Filtering

Calculate a rating matrix from a subset of its entries.

Building a Recommendation Engine with Spark,
https://mapr.com/ebooks/spark/08-recommendation-engine-spark.html

User/ Item	Movie A	Movie B	Movie C	Movie D
Alice	?	4	?	3
Bob	3	?	2	?
Carol	?	5	?	2
Dave	?	?	4	?

Alice	1.4	0.9
Bob	1.2	1
Carol	1.5	0.9
Dave	1.2	0.8

*

A	B	C	D
1.4	1.3	0.9	1.2
0.8	1.1	2	0.8

Alternating Least Squares (ALS)

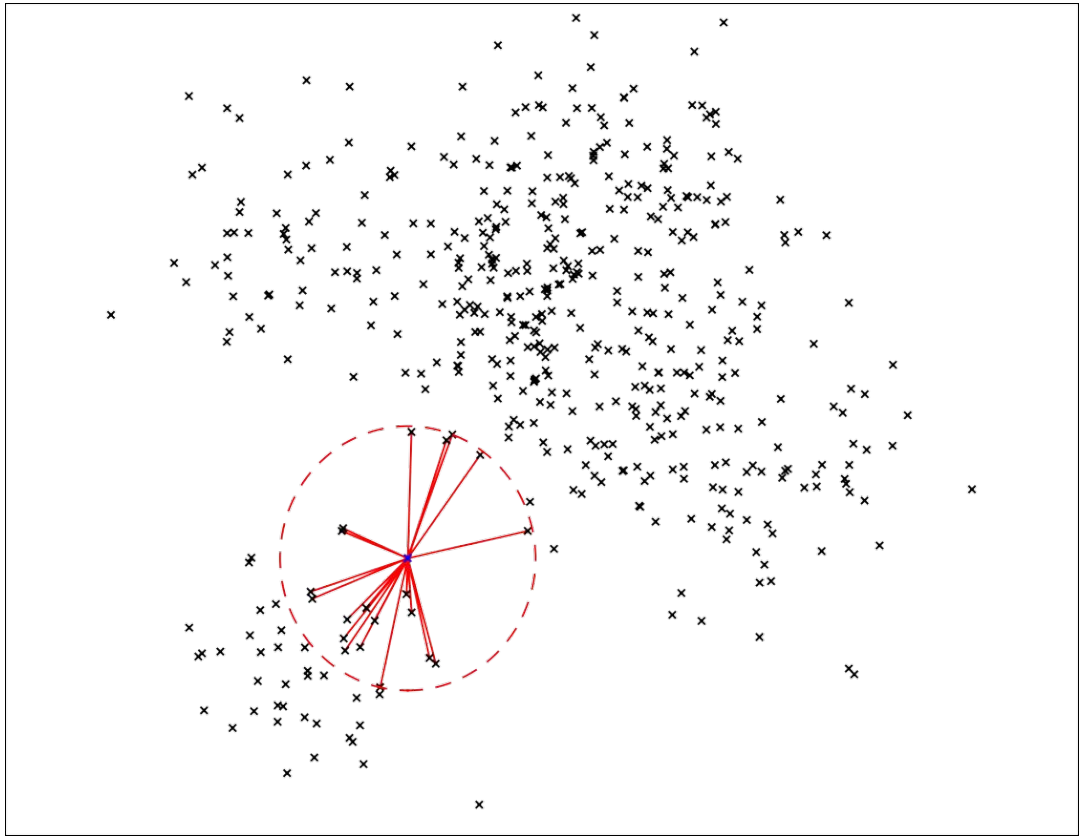
Bulk Synchronous Parallel (BSP) System. Algorithm currently serves as the collaborative filtering method in Apache Spark machine learning library – Spark MLlib

Zhou, Yunhong, et al. "Large-scale parallel collaborative filtering for the netflix prize." International conference on algorithmic applications in management. Springer, Berlin, Heidelberg, 2008.

Das, Ariyam, et al. "Collaborative Filtering as a Case-Study for Model Parallelism on Bulk Synchronous Systems." Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. CIKM, 2017.

Project D
Approximate Nearest Neighbor Search

Annoy - Approximate Nearest Neighbors at Spotify



Further Readings

Fu, Cong, et al. "Fast approximate nearest neighbor search with the navigating spreading-out graph." Proceedings of the VLDB Endowment 12.5 (2019): 461-474.

Dong, Wei, Charikar Moses, and Kai Li. "Efficient k-nearest neighbor graph construction for generic similarity measures." Proceedings of the 20th International Conference on World Wide Web. WWW, 2011.

https://github.com/spotify/annoy